Securing Distance-Vector Routing Protocols

Bradley R. Smith
Computer Sciences
University of California
Santa Cruz, CA 95064
brad@cse.ucsc.edu

Shree Murthy
Sun Microsystems, Inc.
2550 Garcia Avenue
Mountain View, CA 94043
shree@eng.sun.com

J.J. Garcia-Luna-Aceves
Computer Engineering
University of California
Santa Cruz, CA 95064
jj@cse.ucsc.edu

Abstract

We analyze the security requirements of distance-vector routing protocols, identify their vulnerabilities, and propose countermeasures to these vulnerabilities. The innovation we propose involves the use of mechanisms from the path-finding class of distance-vector protocols as a solution to the security problems of distance-vector protocols. The result is a proposal that effectively and efficiently secures distance-vector protocols in constant space.

1. Introduction

Routing protocols dynamically configure the packet forwarding function in internets which allows for the continued delivery of packets in spite of changes in network topology and usage patterns. These changes typically occur due to the ongoing introduction, failure, and repair of network links and routing nodes, which the protocols have been designed to accommodate. The compromise of the routing function in an internet can lead to the denial of network service, the disclosure or modification of sensitive routing information, the disclosure of network traffic, or the inaccurate accounting of network resource usage.

Current routing protocols contain few, if any, mechanisms to provide for the security of their operation. Those that exist are often incomplete. For example, the security mechanisms currently defined for BGP[15] and RIPv2 [8] protect the transmission of routing messages across local networks; however, they do not provide integrity or authenticity of the routing information itself as it traverses an internet. These mechanisms require trust of neighbors regarding updates describing the full internet and, transitively, similar trust of all routers in an internet. More recent efforts are addressing both the security of routing message transmission and of the routing information itself; however, they target link-state protocols [11]. While this class of routing protocols has the advantage of a more straightforward means of securing routing information in a manner that effectively limits the scope of trust, it also involves considerable computation and space overhead that compromise their usability in large-scale internets. Given the evolution of the global Internet to a commercial, production network infrastructure, this state of affairs is clearly unacceptable.

Perlman has defined two classes of network failures [14]. A *simple failure* occurs when a node or a link in a network becomes inoperative, and ceases to function at all. A *Byzantine failure*, defined in terms of the “Byzantine Generals Problem” [7], occurs when nodes or links continue to operate, but incorrectly. A node with a Byzantine failure may corrupt, delay, or forge messages, or may send conflicting messages to different nodes. Possible causes of Byzantine failures include mis-configured nodes, software bugs, unusual hardware failure modes, and hostile attacks. In addition, she defines classes of network robustness. *Simple robustness* is the resistance to malfunctions caused by simple failures. *Byzantine robustness* is the ability to continue correct operation in the presence of arbitrary nodes with Byzantine failures. Modern routing protocols are designed to provide simple robustness. In this context our goals are to provide:

- Byzantine robustness to faults in non-routing nodes in an internet,
- Byzantine Robustness to faults in routing nodes concerning links not incident on those nodes, and
- simple robustness of all other faults in routing nodes.

Section 2 analyzes the security of distance-vector algorithms, and identifies their vulnerabilities and the threats to which they are susceptible. Section 3 presents our proposed strategies and countermeasures for securing distance-vector routing algorithms. Section 4 reviews related work. Section 5 presents our conclusions.

2. Vulnerabilities in distance-vector protocols

The proposed Internet Security Architecture (ISA) [16] provides an architecture for the inclusion of security facilities in the design of protocols to be used in the Internet. Fundamental to the ISA are four concepts:

Vulnerability: A weakness in a system's security that may be exploited by an intruder.

Threat: A potential violation of security. Requires an intruder who has the capability to exploit an existing vulnerability. Threats can be classified into four general categories. *Disclosure* is an event in which an entity gains access to data that the entity is not authorized to receive. *Deception* is an event that results in an authorized entity receiving false data and believing it to be true. *Disruption* is an event that interrupts or prevents the correct operation of system services or functions. And, *usurpation* is an event that results in control of system services or functions by an unauthorized entity.

Security Service: Vulnerabilities and threats are minimized or eliminated through the provision of six security services [13]. *Confidentiality* is the protection of data so it is not made available or disclosed to unauthorized individuals, entities, or processes. *Integrity* is the protection of data so that it is not altered or destroyed in an unauthorized manner. *Authenticity* is the verification of the identity claimed by a system entity. *Access Control* is the protection against unauthorized use of system resources. *Non-Repudiation* is the protection against false repudiation of a communication. *Availability* is the assurance that resources are accessible and usable upon demand by an authorized entity.

Countermeasure: A mechanism or feature that provides a security service. Examples of countermeasures include encryption of network traffic to provide confidentiality, and the use of challenge-response technology for providing authentication of user logins. The cryptographic tools we will use to implement countermeasures to routing protocol vulnerabilities are primarily encryption and digital signatures.

We now use these concepts to develop a solution for securing distance-vector routing protocols using the following methodology:

1. Analyze the protocol design to identify vulnerabilities and threats.
2. Identify the security services needed to reduce or eliminate the vulnerabilities.
3. Design the appropriate countermeasures to provide the needed services.

We consider only threats to the flow of routing traffic, and do not address threats to the flow of data traffic. We describe attacks in terms of different classes of internet nodes, including: authorized routers, and intruders. Authorized routers are those nodes intended by the authoritative network administrator to participate in the routing dialog and computation, running correct and bug-free code, and using correct and bug-free configuration information.

2.1. Intruders

We assume that intruders can position themselves at any point in the network through which all traffic of interest flows, and that an intruder has the capability to fabricate, replay, monitor, modify, or delete any of this traffic. Interpreting this description for a routing environment, we identify the following general classes of intruders:

Masquerading routers: A masquerading router occurs when a node successfully forges an authorized router's identity. This can be accomplished using the IP spoofing [9], or source routing attacks.

Subverted routers: A subverted router occurs when an authorized router is caused to violate the routing protocols, or to inappropriately claim authority for network resources. This typically occurs due to bugs in the routing code, mistakes in the configuration information, or by causing a router to load unauthorized software or configuration information. The specifics of how this can occur depends on the design and configuration of the router.

Unauthorized routers: An unauthorized router can occur when a node on an internet that is not authorized as a router manages to circumvent any access control mechanisms in place, and participate in the routing dialog and computation. How this is achieved depends on the design and configuration of existing access control mechanisms.

Subverted links: A subverted link occurs when an intruder gains access to the physical medium (e.g. copper or fiber optic cable-plant, the "air-waves", or the electronics used to access them) in a manner that allows some control of the channel.

2.2. Threats to routing information

There are a number of vulnerabilities that allow a strategically placed intruder to fabricate, modify, replay, or delete routing traffic. With these capabilities an intruder can compromise the network in a number of ways. The modification or fabrication of routing updates allows an intruder to reconfigure the logical routing structure of an internet, potentially resulting in the denial of network service, the disclosure of network traffic, and the inaccurate accounting of network resource usage. The replay or deletion of routing updates blocks the evolution of subsets of the logical routing structure (in response to topological or link cost changes), or resets it to an arbitrary earlier configuration with potential results similar to above. The vulnerabilities these attacks exploit is the lack of access control, authentication, and integrity of routing messages.

In addition, it is relatively easy for an intruder to gain access to routing traffic. The information carried in this traffic describes the next hop to take to reach a destination. This information is available from other sources, such as monitoring authorized traffic to the desired destination for the next hop it uses, and therefore cannot be protected by measures only involving the routing traffic. Additionally, in pathfinding protocols, the routing traffic includes information describing the path taken by traffic to different destinations. In some circumstances this information may be considered confidential. Since the only source of this information is the routing protocol, it should be possible to protect with modifications to the routing protocols only. The vulnerabilities these attacks exploit are the lack of confidentiality of peer links, and the level of trust placed in routers.

3. Distance-Vector protocol security countermeasures

Fundamentally, there are two classes of communication occuring in routing protocols:

- Communication between neighboring routers, composed of routing updates for destinations the sender has determined are appropriate to forward to the receiver.
- Communication between a given router and an arbitrary set of remote routers, dynamically determined by routing decisions, composed of the fields of routing updates which describe a given destination.

Correspondingly, we present two classes of countermeasures, described in terms of distance-vector algorithms:

- routing message protection countermeasures, and
- routing update protection countermeasures.

Figure 1 shows the additional information used by our countermeasures, and Figure 2 specifies the procedures used to secure a distance-vector routing protocol. We make a number of assumptions in designing these security mechanisms for distance-vector protocols.

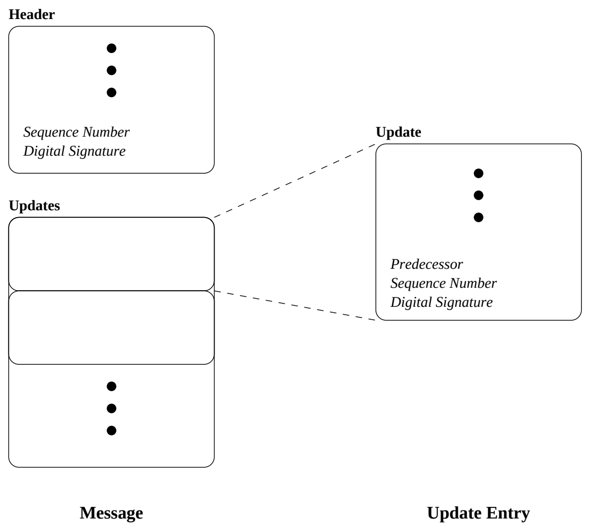


Figure 1. Proposed Routing Message Changes

- We assume intruders have the capabilities described in Section 2.1.
- We assume that a router can trust information it receives from other routers only concerning links incident on the remote routers.
- We assume each router is assigned a public-key pair for use in digitally signing routing messages.
- We assume that key distribution is based on domain names, and that domain names can be efficiently and securely determined given an IP address of a host. The Domain Name System with the proposed security extensions [4] might meet these requirements.

3.1. Routing message protection countermeasures

The following countermeasures are effectively implementing security services not available from lower level transport or network protocols. Specifically, the routing message digital signature and sequence numbers are providing authentication and integrity services of routing messages, which compose the first class of communication described above. If these services were available from network [1, 2] or transport layer protocols, these mechanisms would no longer be needed in the routing protocols.

A number of data structures are defined for use in the pseudo-code for the distance-vector algorithms defined below.

Sequence Number ($seqNum$): The sequence number maintained by each router.

Sequence Number Table (SN^i_{jm}): The Sequence Number table maintained at node i contains the largest sequence value seen in a routing update with originating speaker m for destination network j.

Link-Cost Table (L_i): The Link-Cost table maintained at node i describes the networks node i is attached to. Each entry includes the following information:

- $l^i_n \rightarrow$ the cost of the link to the attached network n. The cost of a failed link or a link to a failed network is infinity.
- $lmod^i_n \rightarrow$ a boolean indicating whether this entry has been modified.

Distance Table (DT_i): The Distance Table at speaker i is a matrix containing, for each destination network j and neighboring speaker k, the following information regarding the route reported by k:

- $D^i_{jk} \rightarrow$ distance from k to j.
- $p^i_{jk} \rightarrow$ predecessor network.
- $sn^i_{jk} \rightarrow$ update sequence number.
- $id^i_{jk} \rightarrow$ identifier of the originating speaker.
- $ds^i_{jk} \rightarrow$ digital signature of the destination, predecessor, and sequence number information as computed by j.

Routing Table (RT_i): The Routing Table at speaker i is a column vector of entries for each known destination network j which specify the following regarding the routes chosen by i:

- $D^i_j \rightarrow$ distance from i to j.
- $p^i_j \rightarrow$ predecessor network.
- $s^i_j \rightarrow$ successor speaker.
- $sn^i_j \rightarrow$ update sequence number.
- $id^i_j \rightarrow$ identifier of the originating speaker.
- $ds^i_j \rightarrow$ digital signature of the destination, predecessor, and sequence number information as computed by j.
- $RT\,mod^i_j \rightarrow$ a boolean indicating whether this entry has been modified.

Update Message (U_k): Each update, U_k, from received by speaker i from neighboring speaker k is a column vector of update entries with the following fields:

- $j \rightarrow$ destination.
- $UD^k_j \rightarrow$ distance from k to j.
- $up^k_j \rightarrow$ predecessor network.
- $usn^k_j \rightarrow$ update sequence number.
- $uid^k_j \rightarrow$ identifier of the originating speaker.
- $uds^k_j \rightarrow$ digital signature of the destination, predecessor, and sequence number information computed by j.

In addition a number of routines are called in the pseudo-code, but not defined.

DigSig(j, p, sn)$_x$: This routine returns the digital signature of the destination network j, predecessor router p, and sequence information sn using the key specified by x. The specific digital signature algorithm used depends on the specifics of the particular implementation.

DTPred(i, j, k, p): This routine finds the predecessor, p, on the path from speaker i to destination network j as reported by neighboring speaker k as recorded in the Distance Table. The specifics of how this is done depend on the particular implementation.

SelectRoute(i, j): This routine picks the preferred route from speaker i to destination network j among the available routes with the highest sequence number. The specifics of how this decision is made depends on the particular implementation.

Network(x): This routine returns the attached network from L_i that is shared with neighboring speaker x.

TransmitUpdate(k, U): This routine transmits the update U to neighbor k.

```
procedure LinkChange($i$, $n$, $c$)
when router $i$ detects a change of its link to network $n$ to cost $c$
begin
    $l^i_n \leftarrow c$;
    $lmod^i_n \leftarrow$ TRUE;
    call UpdateRT($i$);
end

procedure ReceiveUpdate($U_k$)
when router $i$ receives an update $U_k$ from router $k$; assume $U_k$ contains updates in order of dependency
begin
    for each update entry ($j$, $UD^k_j$, $up^k_j$, $usn^k_j$, $uid^k_j$, $uds^k_j$) in $U_k$ do
    begin
        if ValidateUpdate($i$, $j$, $k$, $up^k_j$, $usn^k_j$, $uid^k_j$, $uds^k_j$)
        then begin
            $D^i_{jk} \leftarrow UD^k_j$; $p^i_{jk} \leftarrow up^k_j$; $sn^i_{jk} \leftarrow usn^k_j$; $id^i_{jk} \leftarrow uid^k_j$;
            $ds^i_{jk} \leftarrow uds^k_j$; $SN^i_{j,uid^j_k} \leftarrow usn^k_j$;
        end
        else error "Invalid Update"
    end
    call UpdateRT($i$);
end

/* Validate the update for destination $j$ received by $i$ from neighbor $k$. */
/* Specifically verify sequence number, update entry digital signature, and the path from $j$. */
function ValidateUpdate($i$, $j$, $k$, $p$, $sn$, $id$, $ds$) → boolean;
begin
    if ($sn < SN^i_{j,id}$) then return FALSE;
    if ($ds \neq$ DigSig($j$, $p$, $sn$)$_{id}$) then return FALSE;
    repeat
        $p \leftarrow$ DTPred($i$, $j$, $k$, $p$);
    until (($p =$ FAIL) $or$ ($p$ $in$ $L_i$));
    return ($p$ $in$ $L_i$);
end

procedure UpdateRT($i$)
begin
    for each destination $j$ in $DT^i$ do
    begin
        ($D^i_{jx}$, $p^i_{jx}$, $sn^i_{jx}$, $id^i_{jx}$, $ds^i_{jx}$) ← SelectRoute($i$, $j$);
        if (($D^i_j \neq D^i_{jx}$) $or$ ($s^i_j \neq x$) $or$ ($ds^i_j \neq ds^i_{jx}$))
        then begin
            $D^k_j \leftarrow D^i_{jx} + l^i_{Network(x)}$; $s^i_j \leftarrow x$; $p^i_j \leftarrow p^i_{jx}$; $sn^i_j \leftarrow sn^i_{jx}$;
            $id^i_j \leftarrow id^i_{jx}$; $ds^i_j \leftarrow ds^i_{jx}$; $RT\,mod^i_j \leftarrow$ TRUE;
        end
    end
    call SendUpdates($i$);
end

procedure SendUpdates($i$)
begin
    for each destination $j$ in $RT_i$ where $RT\,mod^i_j =$ TRUE do
    begin
        $U_{tmp} \leftarrow U_{tmp} \cup$ ($j$, $D^i_j$, $p^i_j$, $sn^i_j$, $id^i_j$, $ds^i_j$);
        $RT\,mod^i_j \leftarrow$ FALSE;
    end
    for each attached network $j$ in $L_i$ do
        call TransmitUpdate($U_{tmp}$);
    $sn \leftarrow seqNum$; $seqNum \leftarrow seqNum + 1$;
    for each attached network $j$ in $L_i$ where $lmod^i_j =$ TRUE do
    begin
        for each attached network $p$ in $L_i$ where $p \neq j$ do
            call TransmitUpdate($p$, ($j$, $l^i_j$, $p$, $sn$, $i$, DigSig($j$, $p$, $sn$)$_i$));
        $lmod^i_j \leftarrow$ FALSE;
    end
end
```

Figure 2. Pseudo-code for Generic Secure Distance-Vector Processing

Routing message sequence number. A sequence number is included in each routing message. This sequence number is initialized to zero on the initialization of a newly booted router, and is incremented with each message. On detection of a skipped or repeated sequence number a reset of the session is forced by the re-initialization of the routing process. The size of this sequence number is made large enough to minimize the chance of it's cycling back to zero. However, in the event that it does, the session is reset by the re-initialization of the routing process.

Routing message digital signature. Each routing message is digitally signed by the sender. This provides authenticity and some degree of integrity (protection from message modification, but not from replay) of the routing dialog. On detection of corruption, the message is dropped.

3.2. Routing update protection countermeasures

The countermeasures presented in this section protect communication between a given router and a set of remote routers, composed of routing update fields set by the originating router that describe a specific destination. These countermeasures provide authenticity and integrity of this communication; confidentiality of this communication is pointless as the potential recipients include all authorized routers in an internet.

Add sequence information to updates. Sequence information is added to each update to protect against the replay of old routing information. This sequence information can be in the form of a sequence number or a timestamp. New sequence information is generated for each route output from the routing selection process. While a number of updates may be generated for each route (one message per peer of the originating speaker), only one sequence number or timestamp is used for all of them. This is necessary as a remote speaker may receive the same route as a number of updates, each describing the same destination but representing different paths; all of these updates must be considered valid. This implies that updates for a given destination must be considered valid if their sequence information is greater than or equal to the current sequence information. Note that sequence information must be maintained and validated on a per router basis. An invalid update is silently dropped.

Add predecessor information to updates. A routing-table update of a distance-vector routing protocol consists of one or multiple entries, each specifying a destination and a distance to the destination. [1] To verify the integrity and authenticity of a given update entry, the router processing the update must make sure that the distance to a destination reported in the update entry corresponds to a path that is valid and authentic for each of its hops. By including the information about the second-to-last hop (predecessor) in the path to a destination, the validity and integrity of the entire path from the router verifying the update to the destination can by verified iteratively using information reported by the routers directly adjacent to the destination and routers immediately adjacent to each intermediate hop in the path. The method used to accomplish this without including complete path information is based on the path-traversal technique of path-finding algorithms.

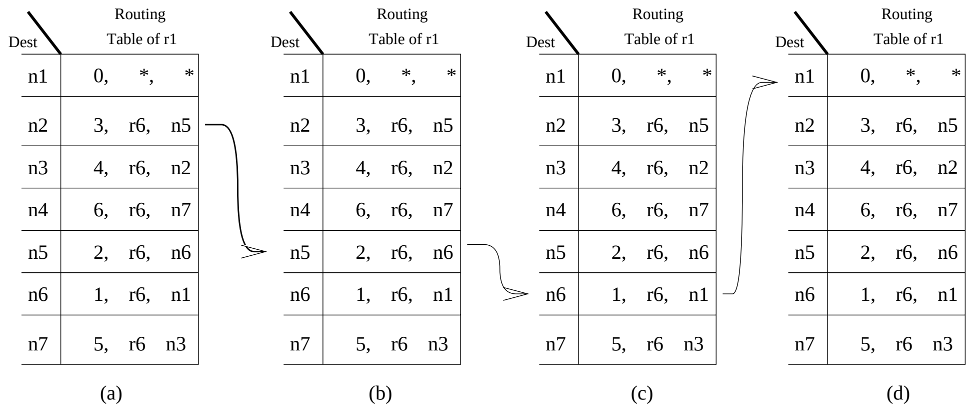
Path-finding algorithms are distance-vector routing algorithms which perform route computation on a per-destination network basis. They maintain information about the next-to-last network along with the distance information from each neighbor to every destination in the internet. This information along with the next hop (successor) to each destination is used to compute loop-free paths to all destinations. Using the information about the next-to-last network (predecessor), an implicit path to a destination can be inferred and thus routing loops can be detected. Path finding algorithms thereby eliminate the well known counting-to-infinity problem of DBF [3]. Each distance and routing table entry is associated with the predecessor information. The design of the path-finding algorithm is such that at all times, the distance and routing table entries satisfy the following property:

> *The path implicit in a distance entry from router i to destination j through a neighbor k, D^k_{ij}, with associated predecessor $h^k_{ij} = h$, is the path implicit to network h, D^k_{ih}, augmented by link (h, j).*

If each column in the distance and routing tables of a router satisfies this property at all times, then it can be used to maintain simple paths to destinations.

Figure 3 illustrates the path traversal using predecessor information. Let $r1$ and $r6$ be the routers, and $n1$–$n7$ be the networks in an internet. The figure shows the routing table entries at node $r1$. A routing table is a vector with each entry specifying the destination j, current shortest distance D^i_j, successor s^i_j and the predecessor p^i_j. Infinite distance is represented as ∞ and null path by *. Let node $r1$ want to determine if a network $n7$ is in the shortest path to destination $n2$. Node $r1$ starts the trace from the entry for destination $n2$ (Figure 3(a)) and finds that the predecessor to $n2$ is network $n5$. Subsequently, $n1$ walks through the predecessors of its path to $n5$ and $n6$ until it reaches the directly connected network $n1$ (Figure 3(d)). From this, node $r1$ determines $n7$ is not in the path from $r1$ to $n2$ (not encountered during the trace). The sequence of predecessors encountered during such a trace represents a path from $n1$

[1] In RIPv2, the successor to the destination is also included.



Dest	Routing Table of r1
n1	0, *, *
n2	3, r6, n5
n3	4, r6, n2
n4	6, r6, n7
n5	2, r6, n6
n6	1, r6, n1
n7	5, r6 n3

Figure 3. Path Traversal using Predecessor Information

to $n2$. This is referred to as the *implicit path* or the path extracted from the predecessor network information.

Each node running the path-finding algorithm maintains a *distance table*, a *routing table* and a *link-cost table*. The distance table at node i is a matrix containing the distance and the predecessor entries for all destinations through all its neighbors. The routing table is a column vector of minimum distance to each destination and its corresponding predecessor and successor information. The link-cost table lists the cost of each link adjacent to the node. The cost of a failed link is considered to be infinity.

Update messages are exchanged between neighbors to update the routing information. An update message contains the source node and the destination network identifiers, and the distance and predecessor information for one or more destinations. The working of algorithms of this type is described by Murthy and Garcia-Luna-Aceves [12, 5].

To secure distance-vector protocols we include the predecessor information defined above in each update. Using this information we then perform a path traversal for each selected route, verifying the integrity of the path and the distance reported for the route.

Digitally sign updates. To ensure the authenticity and integrity of the information used above, the originating router digitally signs the unchanging fields of each update it generates. Specifically, the digital signature covers the *destination, predecessor,* and *sequence information* fields. The distance field of the update is not included because it changes as it propagates through the network. In addition, to allow receiving routers to validate the signature, an IP address of the originating router must be added to each update. These signatures are used to validate a candidate path to a destination before that path is selected for use. This path validation is performed similarly to the loop-detection path traversal specified by the path-finding algorithm, except that the update corresponding to each hop is validated using the digital signature.

3.3. Countermeasure effectiveness

We now analyze the impact of each countermeasure on the threats described in Section 2. In the following we assume the digital signature countermeasure includes facilities for secure authentication and access control.

In broad terms, the message protection countermeasures provide protection against all nodes which lack the necessary cryptographic keys, specifically unauthorized routers, masquerading routers, and subverted links. The digital signature of routing messages protects them from fabrication, modification, and disclosure by these classes of intruders. The addition of a sequence number to routing messages protects them from replay or deletion by these intruders.

Similarly, the update protection countermeasures provide protection against the compromise of those nodes that do have the cryptographic keys, specifically subverted routers. The digital signature of each update protects them from fabrication or modification by subverted routers. The addition of a sequence number to each update protects against replay by a subverted router. The addition of predecessor information to each update provides a means of validating a link in the internet, which can then be used with the routing table to validate the implicit path of an route.

There were a few vulnerabilities we did not address. Specifically, a subverted router is still able to fabricate destination information, delete routing updates, and disclose routing information. The fabrication of destination information requires operational changes in addition to protocol changes to protect against; for example it could be required that a routing authority sign destination information with the name of the originating router to allow recipients to verify that the originating router is connected to the destination. Vulnerability to the deletion and disclosure of routing updates is inherent in the requirements of routing protocols to trust routers in their handling of routing updates. It is likely, due to the high degree of connectivity in most operational internets, that the deletion of routing updates will be at worst ineffective in cutting off access to destinations, and at best detectable through the correlation of received routing information. Further research is needed into the possibility of detecting such intrusions.

In addition, it is still possible for any class of intruder to disclose routing information. Due to the multicast nature of these protocols we are unable to address the threat of the disclosure of routing messages in an efficient manner. To provide this protection would require replacing the routing message digital signature with encryption of the routing message. However, since each message is received by a number of routers this would require sending out a copy of each update encrypted for each recipient router. This requires a significant change in the protocol, and invokes a significant additional cost in both traffic and CPU time for the encryptions. Further research is needed in this area.

3.4. Cost analysis

The costs for these countermeasures are in the space for new fields, in time for computing the new fields, and in time for performing the path-traversal. Following is a rough summary of these costs. In this summary the assumptions are made that digital signatures are 128 bits, and that a 32 bit timestamp provides an adequate lifetime for key changes in routers.

Space per message: Each routing message grows by a 128 bit digital signature, and a 32 bit timestamp. This is comparable to security mechanisms currently proposed for some protocols (e.g., RIPv2).

Space per update: Each routing update entry grows by a 128 bit digital signature, a 32 bit timestamp, and a 64 bit predecessor field (assuming IPv4 IP addresses plus a network mask).

Time per message: A digital signature must be computed once for each routing message generated by a router, and verified once per receiving router.

Time per update: The predecessor field of an update differs for each interface of the originating router it is sent over. Therefore, the digital signature of an update must be computed once for each link of the originating router. Conversely, each update digital signature is verified once by each router which selects one or more routes whose implicit paths include the link represented by the update.

Time per destination: Each selection of a new path to a destination requires a path-traversal. The frequency such changes is dependent on network topology and link change events. However, as has been demonstrated by Garcia-Luna-Aceves and Murthy, efficient path traversal algorithms add minimal overhead [5].

Note that all of the validation actions, which will likely account for a large share of these costs, can be done on a statistical basis to contain time costs.

4. Related work

Kumar [6] analyzes the security requirements of network routing protocols, and discusses the general measures needed to secure the distance-vector and link-state routing protocol classes. He identifies two sources of attacks: subverted routers, and subverted links. Since attacks by subverted routers are seen as difficult to detect, and of limited value to the intruder, he focuses his attention on securing protocols from attacks by subverted links. For distance-vector protocols this translates to the modification or replay of routing updates. The specific countermeasures he proposes are neighbor-to-neighbor digital signature of routing updates, the addition of sequence numbers and timestamps to the updates, and the addition of acknowledgments and retransmissions of routing updates. These results are similar to ours with the exception that we explicitly assume the existence of subverted routers, and provide countermeasures to protect against them. We feel this is important as routers are potentially vulnerable to attacks from a number of sources, with potentially catastrophic results from success.

Murphy [10] outlines a solution for securing distance-vector protocols that involves including the information used to select a route, signed by the neighbor it received it from, in the routing update it then signs and transmits to its neighbors. She points out that this requires the validation of a number nested signatures equal to the number of routers in the path. This results in both update size and validation computation time problems as the size of the network grows. These problems result, fundamentally, from the redundant signing of link information for paths that are supersets of paths used to reach destinations traversed in the longer path. We avoid these problems by signing only the component link information, in the form of predecessors, and performing a path traversal to validate full paths. This results in the use of constant space, and significantly reduced computation time.

Smith and Garcia-Luna-Aceves [17] have analyzed the Border Gateway Routing Protocol (BGP) in a manner similar to the analysis presented here. BGP is a member of the path-vector class of routing protocols, which carry full path information in the routing updates to allow loop detection, and the use of non-uniform route selection metrics. The solutions developed for BGP are similar to the ones presented here in that they use the cryptographic protection of the first hop information in the path by the destination (originating) router.

5. Concluding remarks

We have analyzed the security requirements of distance-vector routing protocols. Vulnerabilities were found in the these protocols that could potentially result in the deception or disruption of the route computation, or the disclosure of routing information. We presented countermeasures for these protocols that eliminated or minimized most of these vulnerabilities. We presented measures, similar to existing proposals, that protect routing transmissions across local networks from the masquerading router, unauthorized router, and subverted link classes of intruders. In addition, we proposed a new class of protection mechanisms, based on the use of predecessor information, that protects routing updates as they traverse an internet from subverted routers; these mechanisms effectively limit the scope of trust of remote routers to information regarding links incident on the remote router.

We did not address the vulnerabilities involving misrepresentation by a router of its direct environment, and the disclosure of routing information. The former we see as beyond the scope of usual routing protocol functionality. The latter is resolvable, but at an inappropriate cost for the target environments.

In summary, we show that it is possible to effectively and efficiently secure distance-vector protocols. We accomplish this using the predecessor information specified in the path-finding class of distance-vector protocols.

References

[1] R. Atkinson. IP Authentication Header. RFC 1826, Aug. 1995.

[2] R. Atkinson. Security Architecture for the Internet Protocol. RFC 1825, Aug. 1995.

[3] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, 2nd edition, 1992.

[4] D. E. Eastlake 3rd and C. W. Kaufman. Domain Name System Security Extensions. Internet draft: draft-ietf-dnssec-secext-10.txt, Aug. 1996.

[5] J. J. Garcia-Luna-Aceves and S. Murthy. A Loop-Free Path-Finding Algorithm: Specification, Verification, and Complexity. In *Proceedings IEEE INFOCOM*, Boston, MA, Apr. 1995.

[6] B. Kumar. Integration of Security in Network Routing Protocols. *ACM SIGSAC Review*, 11(2):18–25, Spring 1993.

[7] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals Problem. *IEEE Transactions on Programming Languages and Application Systems*, 4(3):382–401, July 1982.

[8] G. Malkin. RIP Version 2: Carrying Additoinal Information. RFC 1723, Nov. 1994.

[9] R. T. Morris. A Weakness in the 4.2BSD Unix TCP/IP Software. Technical Report 117, AT&T Bell Laboratories, Murray Hill, New Jersey 07974, Feb. 1985. ftp://netlib.att.com/-netlib/att/cs/cstr/117.ps.Z.

[10] S. L. Murphy. Presentation in Panel on "Security Architecture for the Internet Infrastructure". Symposium on Network and Distributed System Security, Apr. 1995.

[11] S. L. Murphy. Digital Signature Protection of the OSPF Routing Protocol. *Proceedings: Symposium on Network and Distributed System Security*, 1996. http://bilbo.usy.edu/sndss/sndss96.html.

[12] S. Murthy and J. J. Garcia-Luna-Aceves. A more efficient path-finding algorithm. In *28th Asilomar Conference*, pages 229–233, Pacific Grove, California, Nov. 1994.

[13] D. Nessett. The Internet Security Architecture. In *Proceedings: Internet Security Workshop*. IEEE, Nov. 1994.

[14] R. Perlman. *Network Layer Protocols with Byzantine Robustness*. Report MIT/LCS/TR 429, Massachusetts Institute of Technology, Oct. 1988.

[15] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). RFC 1771, Mar. 1995.

[16] R. W. Shirey. Security Architecture for Internet Protocols. A Guide for Protocol Designs and Standards. Internet Draft: draft-irtf-psrg-secarch-sect1-00.txt, Nov. 1994.

[17] B. R. Smith and J. J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocol. *Proceedings of Global Internet '96*, Nov. 1996.